Inside Knowledge: Graph-based Path Generation with Explainable Data Augmentation and Curriculum Learning for Visual Indoor Navigation

Daniel Airinei[1] Elena Burceanu[2] Marius Leordeanu[1,2,3]
[1] National University of Science and Technology POLITEHNICA Bucharest, Romania
[2] Bitdefender [3] Institute of Mathematics "Simion Stoilow" of the Romanian Academy, Romania
constantin.airinei.stud.acs.upb.ro eburceanu@bitdefender.com leordeanu@gmail.com

Abstract

Indoor navigation is a difficult task, as it generally comes with poor GPS access, forcing solutions to rely on other sources of information. While significant progress continues to be made in this area, deployment to production applications is still lacking, given the complexity and additional requirements of current solutions. Here, we introduce an efficient, real-time and easily deployable deep learning approach, based on visual input only, that can predict the direction towards a target from images captured by a mobile device. Our technical approach, based on a novel graph-based path generation method, combined with explainable data augmentation and curriculum learning, includes contributions that make the process of data collection, annotation and training, as automatic as possible, efficient and robust. On the practical side, we introduce a novel large-scale dataset, with video footage inside a relatively large shopping mall, in which each frame is annotated with the correct next direction towards different specific target destinations. Different from current methods, ours relies solely on vision, avoiding the need of special sensors, additional markers placed along the path, knowledge of the scene map or internet access. We also created an easy to use application for Android, which we plan to make publicly available. We make all our data and code available along with visual demos on our project site[1].

1. Introduction

Navigation is an essential part of human life, and an increasingly important domain, given the advancements in robotics, and the need for self-navigating cars and robots. Large indoor structures, such as shopping malls or subway stations can be confusing for people who do not frequent them, and wayfinding inside them can be a tedious task, which can be eased using approaches such as the one presented in this paper. Indoor navigation is also a central area of research in aiding people with visual impairments navigate public places with which they are unfamiliar.

Indoor navigation is markedly distinct from its outdoors counterpart, where both humans and machines would rely on GPS for broad localization, and on vision for the actual maneuvering through the environment. Indoor navigation comes with the challenge of having weak access to GPS service[27], due to the building structure. The solution must now rely on vision only, for both local movement (avoiding obstacles, knowing when you are in an intersection), as well as global direction (knowing which way to go in an intersection). However, this is rarely the case for indoor solutions, as it is common to make use of additional equipment, such as special mapping of the scene, setting up markers and special indicators across it. However, relying on custom changes to the scene is undesirable, as a guiding system would ideally be independent from pre-deployed infrastructure[20]. Collection of LiDAR, WiFi, BlueTooth fingerprints and other similar data sources, are also commonly used approaches which will be discussed in depth in Section 2. Additionally, navigation can be achieved by computing direction from camera feed data alone, by methods that localize the user on a known map, and computing orientation based on that information. We consider our method to stand out among these, as we achieve navigation without global localization on a map, using 2D images alone, collected from a consumer grade mobile device. We train a ConvNet to predict the walking direction towards a target from a predefined set, in real-time, based on the camera feed of a mobile device.

We make the following contributions:

1. **Automatic graph-based generation of novel training paths.** We introduce an efficient graph-based method to generate new indoor paths between all possible start and end points, which offers a full set of possibilities for training by combining automatically segments of real video footage.

[1] https://sites.google.com/view/indoor-navigation-by-vision/

2. **Improved learning strategies using explainability and curriculum learning.** Using the Grad-CAM algorithm for explaining which parts of input are responsible for errors, we design effective partial masking of the input at training time, to improve robustness. Combined with curriculum learning, in which hard training cases are added during the last stage of training, these strategies help the model overcome most of its cases of failure.
3. **Novel indoor navigation dataset with automatic direction annotation.** We recorded data in a real-life scenario, by walking around and recording video footage in a relatively large shopping mall, at different times of the year and with different levels of agglomeration. We compute the true ground truth direction on the world map, with an efficient vision-only method from the observed optical flow and monocular depth, both automatically estimated.

2. Related Work

It is common for research regarding indoor navigation to propose methods that are ultimately aimed at being incorporated into a commercial product that can help people, especially those with impaired vision, to navigate their surroundings. As such, a lot of studies also propose a mechanism for user guidance, such as audio instructions or a graphic interface. We also do this through visual cues on the screen in the form of a compass. However, in this section, we will focus on the related work mostly from the perspective of the navigation mechanism, since that is the area to which we bring our main contribution. Outdoor navigation is a related field from which we can import relevant approaches and techniques. Many studies concern outdoor navigation in the context of car driving, either for humans[3, 30, 48] or autonomous driving[7, 14, 28, 34]. Other outdoors related studies focus on pedestrians[41, 46] and UAV's[6, 50]. However, our focus is placed on indoor scenarios, where conditions differ and new techniques must be developed. There are a large number of studies regarding indoor navigation, and several surveys[2, 16, 21, 25, 37] that describe them. These surveys generally differentiate methods by a few fundamental differences, such as type of sensors used, requirement of scene maps, requirement of special markers, etc. We will further discuss these categories and position our approach among them.

Navigation based on special sensors. As mentioned, most indoor navigation relies on special technologies, such as RFID, WiFi networks, BlueTooth, infrared, inertial sensors, accelerometers, and others. While they are effective, they are considerably different from our approach, and we can consider them all to be non-vision based methods.

Of these, we note some that localize the user globally[4, 8, 33] on a known map. Others[1, 12] use sensors without positioning the client on an available plan of the scene, more akin to our method. Zheng *et al.*[51] create a mobile application that navigates the user based on pre-recorded paths. They record WiFi and motion data, and match it algorithmically with the historical data, determining which way a user should be going. Their application displays a static image of the correct path to the user, who then has to orient himself to identify it in his surroundings.

Navigation based on vision. Here we find it relevant to make a distinction between solutions that use consumer grade devices, like ours, and ones that use special equipment that can capture stereo images, or 3D images. Of the latter, there are methods such as that described by Teng *et al.*[45], where the scene is mapped using a 3D camera, and the location of the users is then inferred by using a point cloud similarity matching algorithm. RGBD cameras provide depth data that can be used to perform operations such as SLAM and obstacle detection[17, 22, 23, 29, 31].

Of the methods that use mobile camera footage, a common approach is using preset markers such as color codes or QR codes on the scene[5, 9, 10, 26, 35, 36, 49], which increase the navigation reliability at the cost of an increase in deployment difficulty.

Those that do not use markers, rely on scene recognition to place the user on a map[40]. Karkar *et al.*[24] propose an application in which the user sends images of the scene to a server, that uses deep learning to infer the coordinates of the user, and sends back the position where the images were taken. Fusco *et al.*[18] localize a user based on exit signs captured within images captured by a phone.

Main novelty of our approach: it is self-contained, not requiring access to external devices or setup at test time, or access to a building map. We implemented it on a consumer-grade smart phone, requiring only access to its camera, thus being an affordable, easily deployable solution. In this category, methods such as the one described by Li *et al.*[32] can also be included, but are primarily concerned with robot navigation within a room, as opposed to wayfinding in a large building, a task fundamentally different from the one we work with. Padhy *et al.*[38] propose a similar system that allows a drone to autonomously traverse indoor corridors by classifying frames with flight commands, but mostly for low-level maneuvering purposes, not for reaching a target destination.

3. Proposed method

For helping a person to navigate indoors, it is sufficient to give at every moment and in real-time, one of several high-level directions (e.g. forwards, left, right, slight-right, turn around), which will guide that person towards a desired target location. We approach this task as a classification problem, where each direction belongs to a set of discrete classes. Given a video stream captured in real-time with a mobile device, the system will learn to automati-

cally predict, based on visual input alone, one of the direction classes - without any additional sensor information (e.g. GPS, depth sensor, access to the Internet) or knowledge of the map.

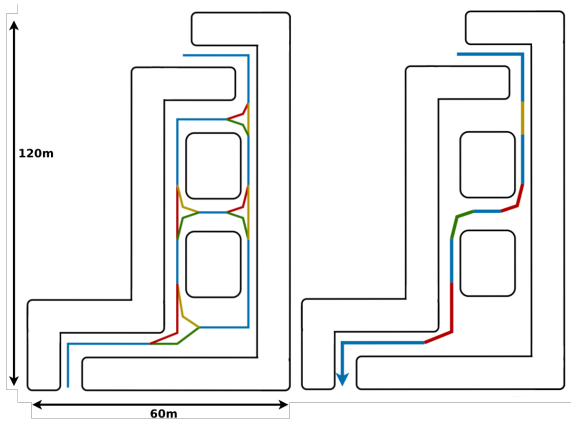

Figure 1. Dataset location floor plan, with visualization of annotated segments (**left**) and generation of a new path that was not ever fully filmed (**right**).

3.1. Data acquisition and preparation

There is currently no published dataset, as required by our approach, which is why we had to collect one. Previous works use simulated environments for similar tasks, but real data is actually needed in order to train, test and develop a robust, real-world application, which is our end practical goal. Thus, we chose an appropriate indoor environment, a medium-sized shopping mall, which is fitted for sufficient data collection for extensive training and testing, during different times of the year, of an indoor navigation system with real practical value. The video footage is recorded with a mobile phone, held at shoulder level, while walking towards different desired destinations, with levels of precision that would be expected of an end user. In some videos the phone is more tilted than in others and the footage usually suffers from shakiness when taking every step. The filming location (general outline can be seen in Figure 1) spans roughly 120 meters in length and 60 meters in width, features 4 intersections and a large number of stores along the corridors that can be used as target destinations. In total, we have around 3 hours of footage filmed along corridors of the mall, with most corridors being visited at least 4 times. To be able to guide a user from one point to another, we have to be able to include all frames of that path at training time, however we found it is not feasible to walk all possible combinations of points in the store. Instead, we exhaustively record footage that contains all possible segments of the mall, annotate them based on a topological map (Figure 1), where intersections are nodes, corridors are edges, and we differentiate ways in an intersection based on the starting and destination corridor. We can then generate any custom path in the mall using these segments, or automatically generate training data as described in algorithm 1.

Algorithm 1 Generate synthetic navigation paths via graph-based video recombination

Require: Graph $G = (V, E)$ where V are intersections and exits (nodes) and E are corridors (edges), annotated video segments for all consecutive triplets, edge weights $w(e)$ as video length (in frames)

Ensure: A valid synthetic path from a random start to a random goal

1: Sample random start node $v_s \in V$
2: Sample random goal node $v_g \in V$
3: Compute shortest path $P = \texttt{Dijkstra}(G, v_s, v_g)$ using edge weights $w(e)$ (minimize total length)
4: Initialize path triplets list $\mathcal{T} \leftarrow []$
5: **for** each consecutive triple (e_i, v_i, e_{i+1}) along path P **do**
6: Retrieve annotated triplet video segment for (e_i, v_i, e_{i+1})
7: Append triplet (e_i, v_i, e_{i+1}) to $\mathcal{T}$
8: **end for**
9: **return** $\mathcal{T}$ as the synthetic path sequence

3.2. Camera motion extraction

For training, we need the ground-truth 3D motion of the camera with respect to the world, which is not trivial to obtain from vision alone, without additional motion sensors (which are often noisy). To get the direction class, during training, we are only interested in the camera rotation around the Y axis, assuming the phone is held almost vertically. To obtain a precise estimation of this rotation we propose a fast and efficient vision-only solution, which combines the observed optical flow, monocular depth estimation and actual 3D camera motion, based on results from visual-based robotics [11], in combination with state of the art optical flow and monocular depth estimation deep networks. Note that this procedure is needed only during the training process and could be performed offline on any system with a GPU. The equation that relates pixel coordinates with respect to the principal point, camera motion, optical flow, and depth is:

$$\begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} -\frac{f}{\rho Z} & 0 & \frac{u}{Z} & \frac{\rho uv}{f} & \frac{f^2+\rho u^2}{f} & \frac{\rho v}{f} \\ 0 & -\frac{f}{\rho Z} & \frac{v}{Z} & \frac{f^2+\rho v^2}{f} & -\frac{\rho uv}{f} & -\frac{\rho u}{f} \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \quad (1)$$

The left-hand side of the equation describes optical flow,

which can be obtained from the video footage. We choose a pretrained deep learning model[15] to do so. The Z parameter is the depth of the scene over the pixel coordinates, which can also be obtained using a pretrained model[47]. An example of the output of these models alongside the original frame can be visualized in Figure 2. The rest of the parameters in the second factor of the equation are the actual pixel coordinates and the intrinsic parameters of the recording device. These are available online for most consumer devices. The only unknown in the equation is the third factor which represents all 6 components of the camera movement (translation and rotation on 3 axis each). We are only interested in ω_y, which will always correlate with the user's orientation throughout the path.

To isolate the rotational component around the Y-axis, from the combined motion represented in Equation (1), we adopt a least squares estimation approach from many N image samples. It can then be interpreted as a linear system of the form:

$$\dot{\mathbf{u}}_i = A_i \mathbf{x},$$

where $\dot{\mathbf{u}}_i = (\dot{u}_i, \dot{v}_i)^\top$ denotes the optical flow vector at the i-th pixel, $A_i \in \mathbb{R}^{2\times 6}$ is the pixel-wise motion matrix determined by depth Z_i, image coordinates (u_i, v_i), and intrinsic parameters, and $\mathbf{x} = (v_x, v_y, v_z, \omega_x, \omega_y, \omega_z)^\top$ is the unknown camera motion vector.

By aggregating N such measurements across randomly sampled pixels, we construct the stacked system:

$$\mathbf{b} = A\mathbf{x},$$

where $A \in \mathbb{R}^{2N\times 6}$ and $\mathbf{b} \in \mathbb{R}^{2N}$ are the concatenations of all A_i and $\dot{\mathbf{u}}_i$ respectively. Given that there are 6 unknowns and each pixel point results in 2 equations, we need at least 3 points to approximate a solution. We solve for $\mathbf{x}$ using the standard least squares solution, by randomly sampling 200 points in the frame for robustness of approximation:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|^2 = (A^\top A)^{-1} A^\top \mathbf{b}.$$

From the estimated motion vector $\hat{\mathbf{x}}$, we extract ω_y as the component of interest, which corresponds to the yaw rotation of the camera and hence the user's heading direction.

3.3. Motion data interpretation

The output of the equation is now a continuous value that is very sensitive to the movement of the phone, as it rotates around the vertical axis with each step. Figure 3 shows the raw data computed from equation (1). The first two plots display the ω_y component, whose values represent camera rotation between any two frames. Positive values indicate a rotation towards the left, and negative values towards the right, while values close to 0 indicated the camera has not rotated at all between two frames. The raw data has noisy

Figure 2. Original frame (**left**). Depth estimation (**center**). Optical flow (**right**)

values which can be smoothed out and processed into discrete values. We convert the values into the 8 classes that represent the rotation between frames. For visualization purposes, it can be considered that these classes correspond to a N/S/W/E compass, that also displays middle points between the cardinal directions. The bottom plot in Figure 3 displays the sequence of classes along the frames of the video. A value of 0 indicates the class that represents the forward movement. The end user should in this situation see a compass displaying an arrow towards north.

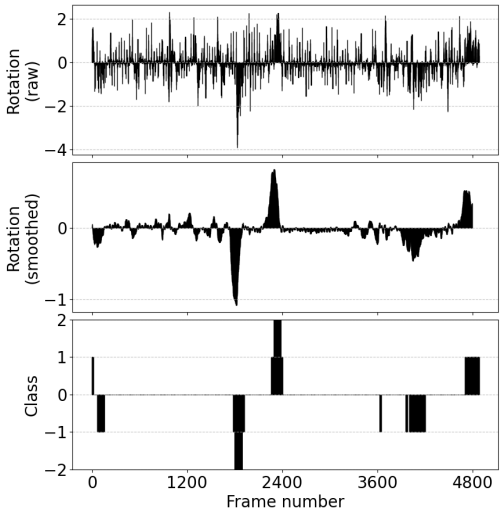

Figure 3. Plots of motion data, where the horizontal axis represents the frame number of the video and vertical axis the rotation component. Raw motion data (**top**), smoothed values (**center**) and conversion into 8 possible discrete values (**bottom**). The 5 peaks visible in the bottom plot correspond to 5 turns in the building in the video. The 3 classes corresponding to backwards movement are not present in this example.

3.4. Model training

For our method, we experimented with deep ConvNets, as they are highly efficient, even in small size, for image classi-

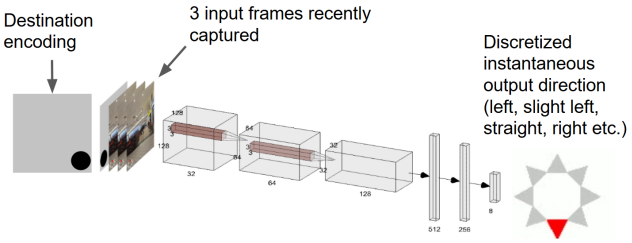

Figure 4. Proposed model architecture. **Input**: 3 frames from recorded footage of what the user sees while navigating and an additional encoded target image. **Model**: 8 million parameter CNN with 3 convolutional and 3 fully connected layers. **Output**: Next motion of the camera, discretized over 8 classes.

fication [19] and can be easily deployed on mobile devices. Our model architecture is shown in Fig. 4. We found in tests that better results are obtained by using 3 RGB frames as input (current one, and two previous ones, 1 second apart). To encode the target destination, a black circle on a white background whose center encodes the target is passed on one of the channels (note that the location encoding is decided in advance, but arbitrary, with no connection to the actual 2D location on the map). Together with the 3 frames, we have a total of 10 channels at input. All images are resized to 128x128 for efficiency before being processed by the network. We also use standard data augmentation to randomly rotate the input images 15° in either direction, to account for the biased rotation a phone held in hand might have. The ground truth for the network is a one hot encoding of 8 classes which represent the predicted direction of walking from a given set of frames towards the target passed as an input. The model (Fig. 4) consists of 3 convolutional layers and 3 fully connected layers, using only ReLU activations, totaling around 8 million learnable parameters. The output of the network is a probability for each of the 8 possible direction classes. For the results presented in this document, we use an Adam optimizer and train the model for at least 10 epochs with a learning rate of 0.001. An important aspect of this task is the unbalanced nature that characterizes the data from large indoor scenes (as also observed in Figure 3): while walking through a mall, or any large building with long hallways, most of the time a user will walk forwards, and only at times will they briefly take a turn to a different direction, if heading straight for a target. Due to this challenge, we chose to use a cross-entropy loss combined with sampling more training frames where the ground truth indicates a left or right turn. We also propose and test more advanced training procedures, to make the model more robust to wrong turns, which we explain later in Sections 4.2.

4. Experiments

The goal of our experimental analysis is to show that the model can correctly associate image frames with walking directions, at every time step (Sec. 4.1, as well as at key moments in intersections or cases of turning around (Sections 4.3, 4.4). We took a more in depth-look at failure cases in Section 4.2 and proposed different masking strategies during training, which along with curriculum learning, addressed the failure cases in large part. Moreover, we also performed tests on out-of-distribution scenarios, and showed that the proposed method is robust to such distribution changes between testing and training cases.

Last but not least, we implemented our system on an Android Smart Phone and tested it in real time in the real world. The application always took the user to the correct place: the rare errors were usually immediately corrected by the system, while the user naturally disregarded directions given when distractors (moving people and obstacles) where covering large parts of the camera input. We will make the application, dataset and training code, publicly available. We also plan to have an extensive evaluation with many users and report their feedback and rate of success in the longer journal version, if accepted.

4.1. Evaluation at every timestep

For a general understanding of the system's ability to learn to give directions from visual input, we first test its performance at every time step in unseen test paths (created from video segments unseen during training) , by using three different evaluation measures: 1) Accuracy: it is the classical measure as percentage of correct decisions among all test cases (as compared to ground truth), along every test path, at every frame. 2) F1 Score: this is the standard F-measure, used such that the positive class includes all decisions that involve a change in direction and the negative class contain all test cases belonging to "move forward" action. The F1-measure, which is designed for problems with significant imbalance between positive and negative classes, is appropriate in this case, where changes in direction (the positive class) are rare compared to the "move forward" class. 3) The Angle Error measure, considers for each class the actual angle, as the center of the class bin around the circle of directions. Then the error is computed as the absolute difference in angle between the estimated direction and the ground truth. This measure is appropriate in order to give a sense of how far the model prediction is from ground truth: for example, a relatively small angle error (difference between right turn and slight-right turn) is less harmful than a large angle error of 180 degrees (right turn vs. left turn).

We test on different types of scenes, with different lengths and level of difficulty based on the number of people present (mostly empty vs. mostly crowded) and number of target destinations. We also tested different training strate-

gies, as explained in the next section 4.2, and present the results in 1. The results show that the presence of people can affect the performance, whereas the system is not sensitive to the length of the path or the number of possible destinations, which is very encouraging. Regarding the negative effect of the presence of people, it is handled by our automatic masking data augmentation strategies proposed and discussed in the next section (with performance presented in Table 1.

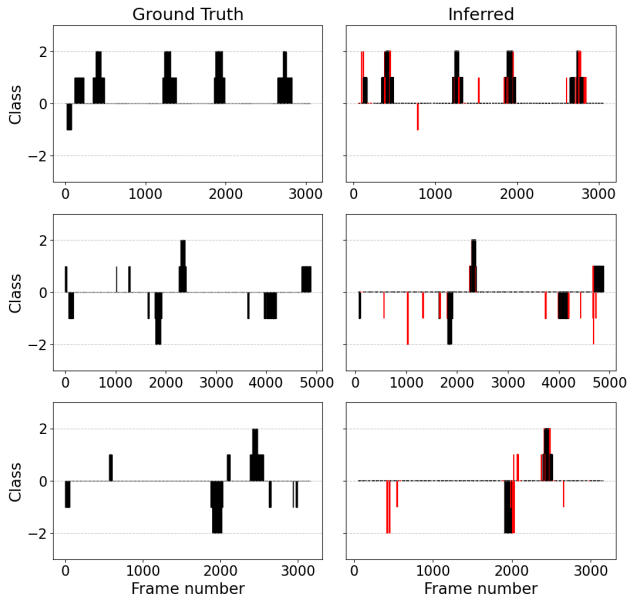

Figure 5. Comparison of real previously extracted motion data (**left**), compared to inferred values (**right**) for three different paths trained separately. Red areas denote wrongfully inferred classes for those frames.

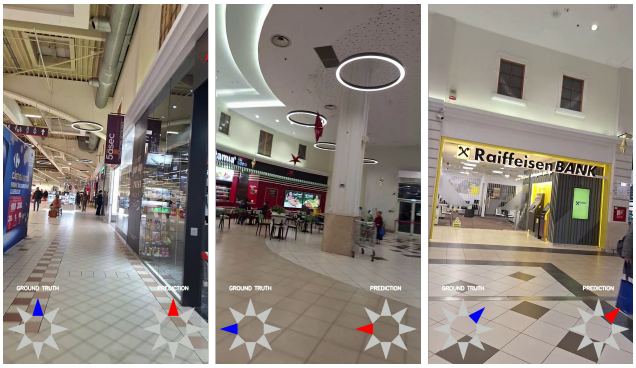

Figure 6. Examples of frames in our application, presenting the guidance compass twice, displaying the ground truth and the predicted value for those specific frames.

4.2. Deeper look into model prediction

We notice that most failure cases take place when there are people right in front of the camera, which can easily explain why the predicted direction could be wrong. Humans also get confused, even in well-known scenes, when nearby obstacles (e,g. other moving people) are obstructing their view. During training, due to the presence of many people in videos, the system implicitly learns to base its output direction both on the final global navigation target and on the local navigation goal of avoiding collision with obstacles and moving people. However, when the moving people are too close to the camera, they occupy a large part of the input image and are expected to confuse the system.

For a more in-depth analysis, we use the guided backpropagation Grad-CAM algorithm[42, 44], that shows which parts of the input image are relevant for the final output of the model. Thus, we can observe which image regions contribute the most to the system's wrong decisions. In Fig. 7 we show the results of Grad-CAM (the input regions relevant for the final decision are shown in stronger red), when making correct vs. wrong decisions. Visual inspection confirms the intuition: wrong decisions are mostly caused by people that are too close to the camera and occupy a relatively large part of the visual field.

These observations led us to design three different data augmentation procedures during training, in order to make the system more robust to such distracting situations (see Fig. 7). First, we introduce the PeopleMask procedure in which we use a person detector[39] and fill the bounding boxes of people in the training set with random noise, and train the model again. With this modification, the network no longer associates direction with people, and instead focuses on static landmarks. Second, we test a random masking strategy (termed RandMask), in which masking boxes are chosen at random locations and sizes, similar to [13] - such a random strategy could make the system more robust by simply creating varied and much more difficult input, in which many details and structures, including people, are randomly cutout. In the third masking approach, we propose a method directly based on the output of the Grad-CAM attention, to make the input even harder during training. Termed GradMask, the idea is to randomly pick masking boxes (which again are filled with random noise) in the activation regions of Grad-CAM. This serves two relevant purposes: 1) it often covers the distracting people in front of the camera, obtaining the same effect as PeopleMask, without the need of using an additional off-the-shelf person detector; 2) it also covers relevant parts of the scene, similar to the second masking strategy, RandMask, thus leading to more robust training. We further extend the last GradMask strategy by adopting a curriculum learning strategy [43], with a second-phase of training in which the model receives mostly hard training cases on which the model failed during

Table 1. Comparison of accuracy, F1 scores, and angle error across various test scenarios, with row-wise averages (corrected).

Method	**Crowded**	**Empty**	**Short paths**	**Long paths**	**Single targets**	**Many targets**	**Average**
Accuracy							
Baseline	0.71	0.92	0.78	0.82	0.75	0.81	0.80
PeopleMask	0.73	0.91	0.81	0.84	0.79	0.82	0.82
RandMask	0.73	0.88	0.79	0.85	0.83	0.84	0.82
GradMask	0.74	0.89	0.80	**0.88**	0.84	0.87	0.84
GradMask + Curriculum	**0.76**	**0.93**	**0.86**	0.87	**0.85**	**0.88**	**0.86**
Average F1 Score							
Baseline	0.48	0.81	0.58	0.65	0.57	0.59	0.61
PeopleMask	0.50	0.8	0.61	0.66	0.59	0.61	0.63
RandMask	0.47	0.76	0.62	0.67	0.59	0.60	0.62
GradMask	0.51	0.77	0.63	**0.70**	0.60	0.62	0.64
GradMask + Curriculum	**0.52**	**0.83**	**0.65**	0.69	**0.61**	**0.63**	**0.65**
Angle Error (degrees)							
Baseline	14	7	11	10	12	12	11.00
PeopleMask	13	8	9	9	10	11	10.00
RandMask	14	10	10	10	11	11	11.00
GradMask	13	8	**8**	**8**	10	8	9.17
GradMask + Curriculum	**12**	**6**	**8**	9	**8**	**7**	**8.33**

the first learning phase.

The extensive experiments, on different type of scenes, in 1, present the performance of the different strategies and also confirm their effectiveness over the baseline (which is trained on data augmented with the graph-based path generation Algorithm 1). Table 1 displays a detailed view of the results, comparing the chosen metrics across our three solutions (the baseline and two dataset augmentations). Generally, placing cutouts across the scene provides better results than covering the people directly, both outperforming the baseline training with no augmentations.

4.3. Evaluation at key moments in intersections

In order to better understand the global capability of the system to guide the user to the correct target, we now look at key moments, in intersections and turn-around situations, where the model has to decide between several paths. If at every such key moment (intersection or turn-around) the right decision is made, then we expect the user to reach the correct target. Thus, we also test the system only on portions of the dataset where the camera approaches an intersection or when the user is facing the wrong way (results in Tab. 2), for two scenarios: 1) when using original video footage along paths, without the training data augmentation using our graph-based path generation approach and 2) with the graph-based generated paths included in the training data: these are paths artificially created by concatenating segments from different videos, collected at different times, in order to create novel paths (not seen during actual collections), towards all possible targets.

Table 2. Average accuracy for intersections and turn–around, with and without our graph-based path generation (Algorithm 1). Note the highly significant performance boost produced by Alg. 1.

Training setup	**Accuracy**	
	Intersection	**Turning around**
Without graph-based path generation	0.52	0.49
With graph-based path generation	**0.89**	**0.91**

When using the graph-based path generation Algorithm 1, the training data is much richer, having virtually all possible paths included, from all starting points to all possible targets. These automatically generated paths, from real video segments, make the training set much larger and richer in terms of combinations of start and end points. Therefore, as the numbers show, the Algorithm is highly effective for improving global navigation performance.

4.4. Evaluation of a special case: Turning around

Sometimes, the target can be situated behind the user such that the best action is to turn around. Because our method naturally trains a model to predict the path of a camera, we tackle this case by training the model using the reversed video footage, such that the correct walking direction is in reverse, towards a specific target. Thus, if the user is ever facing the wrong way, the model will simply predict he must walk backwards, when they should turn around. We trained

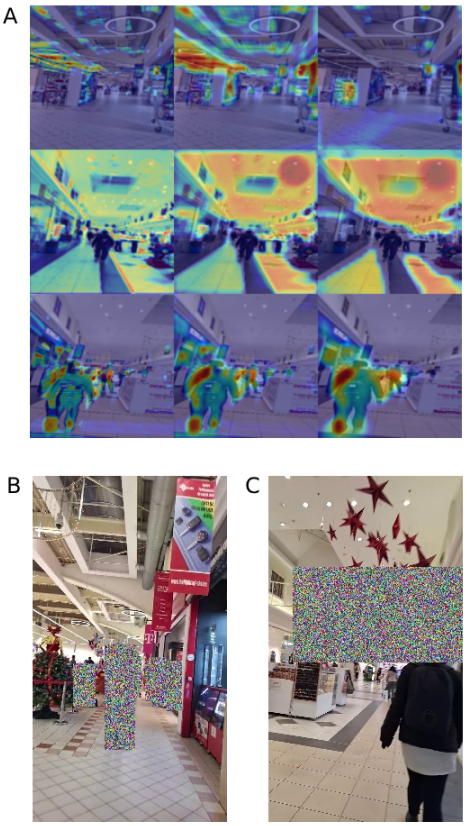

Figure 7. **A.** The network attention computed with Grad-CAM, among the first 3 convolutional layers. The top two rows show the attention when the correct direction prediction is moving forward - usually focused on the ceiling or the floor. The bottom row is the activation of a failure case, when the model predicts a right turn, due to a person being right in front of the camera. **B.** Augmentation based on people detection and removal. **C.** Cutout augmentation using distribution of network attention.

and tested such cases. As shown in Tab. 2, the model predicts with accuracy of 91% the cases when the user should turn around. In practice we expect the effectiveness to be close to 100% because when the system corrects itself after the user takes a few steps.

4.5. Out-of-distribution testing

We further test our method in out-of-distribution cases. In practice the test data can often be captured during a different time of the year from the time of training data collection, so the visual appearance of scenes could be significantly different between the training and testing periods, due to changes in interior decorations, shop facades, signs and people behavior. In Fig. 8 we present such experiments with videos collected during two distinct periods, Christmas and non-Christmas. We tested different combinations of train and test videos, with details explained in Fig. 8.

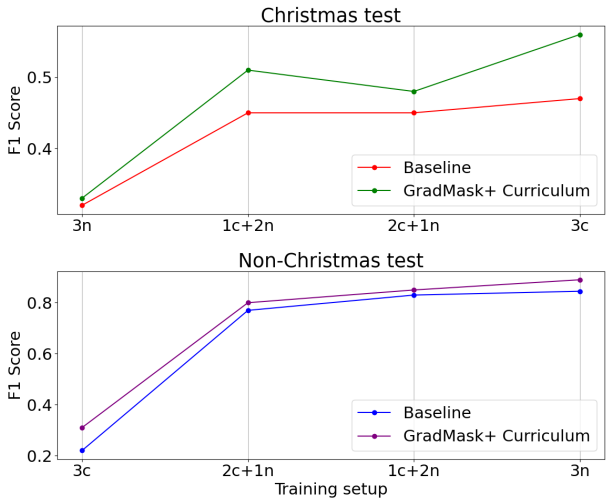

Figure 8. Out of distribution tests: two plots with F1 scores for testing during Christmas period (up) and before (non-) Christmas period (bottom) vs. different combinations of training videos, some filmed during Christmas (labeled as "c") and others filmed before Christmas (labeled as "n"). For example, 1c+2n on the x axis denotes a training set of 1 Christmas video and 2 non-Christmas ones. The visual distribution of scenes during Christmas is drastically different from the other ones, due to the abundance of seasonal decorations and number of people present. Each plot shows that as we increase the amount of in-distribution videos in the training set the performance increases drastically. Interestingly, a single in-distribution video is sufficient in both cases, to reach high performance, showing that the system is robust to a significant amount of out-of-distribution training data, as long as a minimum amount of in-distribution data is present during training.

5. Conclusions

We presented a visual learning approach for indoor mapless navigation with a mobile device, in real time, to predict walking directions in complex, crowded indoor scenes. To our best knowledge, this is the first system of this kind, which, based on vision-only with a consumer-grade smart phone, demonstrates that accurate navigation can be achieved without pre-installed infrastructure, special depth and other navigation sensors, or detailed scene maps. Our technical contributions include a novel graph-based path generation method for producing all possible paths from all start points to all targets, from limited amount of real video footage and different data augmentation techniques, including one based on an explainability model, Grad-CAM, which, combined with curriculum learning, makes the model robust to distractors (crowded scenes of many people walking in front of the camera).

We implemented our system in an easy-to-use applica-

tion for the Android system, which, along with our dataset, we plan to make publicly available, along with all our training code. Future work includes testing in different scenes and out-of-distribution scenarios as well as deploying our application to a larger number of users for a larger scale experimental evaluation.

Acknowledgements

This work is supported in part by projects "Romanian Hub for Artificial Intelligence - HRIA", Smart Growth, Digitization and Financial Instruments Program, 2021-2027 (MySMIS no. 334906) and "European Lighthouse of AI for Sustainability - ELIAS", Horizon Europe program (Grant No. 101120237).

References

[1] A. A. Alajami, G. Moreno, and R. Pous. Design of a uav for autonomous rfid-based dynamic inventories using stigmergy for mapless indoor environments. *Drones*, 6(8), 2022. 2

[2] E. J. Alqahtani, F. H. Alshamrani, H. F. Syed, and F. A. Alhaidari. Survey on algorithms and techniques for indoor navigation systems. In *2018 21st Saudi Computer Society National Computer Conference (NCC)*, pages 1–9, 2018. 2

[3] A. Amini, G. Rosman, S. Karaman, and D. Rus. Variational end-to-end navigation and localization. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8958–8964, 2019. 2

[4] B. Ando, S. Baglio, V. Marletta, and N. Pitrone. A mixed inertial & rfid orientation tool for the visually impaired. In *Proceedings of the 6th International Multi-Conference on Systems, Signals and Devices*, Djerba, Tunisia, March 2009. Held on 23–26 March 2009. 2

[5] S. Anup, A. Goel, and S. Padmanabhan. Visual positioning system for automated indoor/outdoor navigation. In *TENCON 2017 - 2017 IEEE Region 10 Conference*, pages 1027–1031, 2017. 2

[6] M. Y. Arafat, M. M. Alam, and S. Moh. Vision-based navigation techniques for unmanned aerial vehicles: Review and challenges. *Drones*, 7(2), 2023. 2

[7] B. Barua, C. Gomes, S. Baghe, and J. Sisodia. A self-driving car implementation using computer vision for detection and navigation. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pages 271–274, 2019. 2

[8] K. Beydoun, V. Felea, and H. Guyennet. Wireless sensor network system helping navigation of the visually impaired. In *Proceedings of the IEEE International Conference on Information and Communication Technologies: From Theory to Applications*, Damascus, Syria, April 2008. Held on 7–11 April 2008. 2

[9] A. Botta and G. Quaglia. Performance analysis of low-cost tracking system for mobile robots. *Machines*, 8(2):29, 2020. 2

[10] Y. Chang, S. Tsai, and Y. Wang. A context aware handheld wayfinding system for individuals with cognitive impairments. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*, Halifax, NS, Canada, October 2008. Held on 13–15 October 2008. 2

[11] P. Corke. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, volume 73 of *Springer Tracts in Advanced Robotics*. Springer, 2011. 3

[12] A. Corrales-Paredes, M. Malfaz, and M. Salichs. Signage system for the navigation of autonomous robots in indoor environments. *Industrial Informatics, IEEE Transactions on*, 10:680–688, 02 2014. 2

[13] T. Devries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017. 6

[14] T.-D. Do, M.-T. Duong, Q.-V. Dang, and M.-H. Le. Real-time self-driving car navigation using deep neural network. In *2018 4th International Conference on Green Technology and Sustainable Development (GTSD)*, pages 7–12, 2018. 2

[15] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015. 4

[16] N. El-Sheimy and Y. Li. Indoor navigation: state of the art and future trends. *Satellite Navigation*, 2(7), 2021. 2

[17] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the rgb-d slam system. In *2012 IEEE International Conference on Robotics and Automation*, pages 1691–1696, 2012. 2

[18] G. Fusco, S. A. Cheraghi, L. Neat, and J. M. Coughlan. An indoor navigation app using computer vision and sign recognition. In *Computers Helping People with Special Needs*, volume 12376 of *Lecture Notes in Computer Science*, pages 485–494, September 2020. Epub 2020 Sep 4. PMID: 33263114; PMCID: PMC7703403. 2

[19] T. Guo, J. Dong, H. Li, and Y. Gao. Simple convolutional neural network on image classification. In *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, pages 721–724, 2017. 5

[20] I. A. Hashish, G. Motta, M. Meazza, G. Bu, K. Liu, L. Duico, and A. Longo. Navapp: An indoor navigation application: A smartphone application for libraries. In *2017 14th Workshop on Positioning, Navigation and Communications (WPNC)*, pages 1–6, 2017. 1

[21] H. Huang and G. Gartner. *A Survey of Mobile Indoor Navigation Systems*, pages 305–319. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. 2

[22] Q. Jin, Y. Liu, Y. Man, and F. Li. Visual slam with rgb-d cameras. In *2019 Chinese Control Conference (CCC)*, pages 4072–4077, 2019. 2

[23] N. Kanwal. A navigation system for visually impaired: A fusion of vision and depth sensor. *Applied Bionics and Biomechanics*, 2015, 08 2015. 2

[24] A. G. Karkar, S. Al-Maadeed, J. Kunhoth, et al. Camnav: A computer-vision indoor navigation system. *Journal of Supercomputing*, 77:7737–7756, 2021. 2

[25] D. Khan, Z. Cheng, H. Uchiyama, S. Ali, M. Asshad, and K. Kiyokawa. Recent advances in vision-based indoor navigation: A systematic literature review. *Computers and Graphics*, 104:24–45, 2022. 2

[26] D. Khan, S. Ullah, and S. Nabi. A generic approach toward indoor navigation and pathfinding with robust marker tracking. *Remote Sensing*, 11(24):3052, 2019. 2

[27] M. Kjærgaard, H. Blunck, T. Godsk, T. Toftkjær, D. Lund, and K. Grønbæk. Indoor positioning using gps revisited. pages 38–56, 05 2010. 1

[28] A. Kumar, T. Saini, P. B. Pandey, et al. Vision-based outdoor navigation of self-driving car using lane detection. *International Journal of Information Technology*, 14:215–227, 2022. Received: 05 December 2020; Accepted: 15 July 2021; Published: 04 August 2021; Issue Date: February 2022. 2

[29] Y. H. Lee and G. Medioni. Rgb-d camera based wearable navigation system for the visually impaired. *Computer Vision and Image Understanding*, 149:3–20, 2016. Special issue on Assistive Computer Vision and Robotics - "Assistive Solutions for Mobility, Communication and HMI". 2

[30] M. Leordeanu and I. Paraicu. Driven by vision: Learning navigation by visual localization and trajectory prediction. *Sensors*, 21(3), 2021. 2

[31] B. Li, J. P. Muñoz, X. Rong, Q. Chen, J. Xiao, Y. Tian, A. Arditi, and M. Yousuf. Vision-based mobile indoor assistive navigation aid for blind people. *IEEE Transactions on Mobile Computing*, 18(3):702–714, 2019. 2

[32] F. Li, C. Guo, B. Luo, and H. Zhang. Multi goals and multi scenes visual mapless navigation in indoor using meta-learning and scene priors. *Neurocomputing*, 449:368–377, 2021. 2

[33] J. M. Loomis, R. G. Golledge, R. L. Klatzky, and J. R. Marston. Assisting wayfinding in visually impaired travelers. In *Applied Spatial Cognition: From Research to Cognitive Technology*, pages 179–202. Lawrence Erlbaum Associates, Mahwah, NJ, USA, 2006. 2

[34] J. López, P. Sánchez-Vilariño, R. Sanz, and E. Paz. Efficient local navigation approach for autonomous driving vehicles. *IEEE Access*, 9:79776–79792, 2021. 2

[35] R. Manduchi, S. Kurniawan, and H. Bagherinia. Blind guidance using mobile computer vision: A usability study. In *Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility*, Orlando, FL, USA, October 2010. Held on 25–27 October 2010. 2

[36] M. L. Mekhalfi, F. Melgani, A. Zeggada, F. G. De Natale, M. A.-M. Salem, and A. Khamis. Recovering the sight to blind people in indoor environments with smart technologies. *Expert Systems with Applications*, 46:129–138, 2016. 2

[37] A. Morar, A. Moldoveanu, I. Mocanu, F. Moldoveanu, I. E. Radoi, V. Asavei, A. Gradinaru, and A. Butean. A comprehensive survey of indoor localization methods based on computer vision. *Sensors*, 20(9), 2020. 2

[38] R. P. Padhy, S. Verma, S. Ahmad, S. K. Choudhury, and P. K. Sa. Deep neural network for autonomous uav navigation in indoor corridor environments. *Procedia computer science*, 133:643–650, 2018. 2

[39] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015. 6

[40] E. Royer, M. Lhuillier, M. Dhome, et al. Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision*, 74(3):237–260, 2007. Received: 14 October 2005; Accepted: 11 December 2006; Published: 13 January 2007; Issue Date: September 2007. 2

[41] L. Ruotsalainen. *Vision-Aided Pedestrian Navigation for Challenging GNSS Environments*. Suomen geodeettisen laitoksen julkaisuja - Publications of the Finnish Geodetic Institute;151. Suomen geodeettinen laitos, Nov. 2013. Awarding institution:Tampereen teknillinen yliopisto - Tampere University of Technology¡br/¿Submitter:Submitted by Kaisa Kulkki (kaisa.kulkki@tut.fi) on 2013-11-12T10:30:59Z No. of bitstreams: 1 ruotsalainen.pdf: 3916148 bytes, checksum: 50e6a42b22791d2b4dbb832b3b3dec65 (MD5)¡br/¿Submitter:Approved for entry into archive by Kaisa Kulkki (kaisa.kulkki@tut.fi) on 2013-11-13T10:10:53Z (GMT) No. of bitstreams: 1 ruotsalainen.pdf: 3916148 bytes, checksum: 50e6a42b22791d2b4dbb832b3b3dec65 (MD5)¡br/¿Submitter:Made available in DSpace on 2013-11-13T10:10:53Z (GMT). No. of bitstreams: 1 ruotsalainen.pdf: 3916148 bytes, checksum: 50e6a42b22791d2b4dbb832b3b3dec65 (MD5). 2

[42] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016. 6

[43] P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe. Curriculum learning: A survey. *CoRR*, abs/2101.10382, 2021. 6

[44] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net, 2015. 6

[45] X. Teng, D. Guo, Y. Guo, X. Zhou, and Z. Liu. Cloudnavi: Toward ubiquitous indoor navigation service with 3d point clouds. *ACM Trans. Sen. Netw.*, 15(1), Jan. 2019. 2

[46] S. TREUILLET and E. ROYER. Outdoor/indoor vision-based localization for blind pedestrian navigation assistance. *International Journal of Image and Graphics*, 10(04):481–496, 2010. 2

[47] M. Viola, K. Qu, N. Metzger, B. Ke, A. Becker, K. Schindler, and A. Obukhov. Marigold-dc: Zero-shot monocular depth completion with guided diffusion. *arXiv preprint arXiv:2412.13389*, 2024. 4

[48] A. Vu, A. Ramanandan, A. Chen, J. A. Farrell, and M. Barth. Real-time computer vision/dgps-aided inertial navigation system for lane-level vehicle navigation. *IEEE Transactions on Intelligent Transportation Systems*, 13(2):899–913, 2012. 2

[49] A. Zeb, S. Ullah, and I. Rabbi. Indoor vision-based auditory assistance for blind people in semi-controlled environments. In *Proceedings of the 4th International Conference on Image Processing Theory, Tools and Applications*, Paris, France, October 2014. Held on 14–17 October 2014. 2

[50] J. Zhang, W. Liu, and Y. Wu. Novel technique for vision-based uav navigation. *IEEE Transactions on Aerospace and Electronic Systems*, 47(4):2731–2741, 2011. 2

[51] Y. Zheng, G. Shen, L. Li, C. Zhao, M. Li, and F. Zhao. Travinavi: Self-deployable indoor navigation system. *IEEE/ACM Transactions on Networking*, 25(5):2655–2669, 2017. 2